A comparison of statistical learning approaches for engine torque estimation

A. Rakotomamonjy, Rodolphe Le Riche, David Gualandris, Zaid Harchaoui

A comparison of statistical learning approaches for engine torque estimation

A. Rakotomamonjy[1], R. Le Riche[2], D. Gualandris[3] and Z. Harchaoui[4]

[1] LITIS EA 4051 , INSA Rouen, 76801 St. Etienne du R$^{\text{ay}}$ France
[2] CNRS - UMR5146, 3MI and SMS, Ecole des Mines de Saint Etienne, France
[3] PSA, Dir. de la Recherche et de l'Innovation Automobile, Velizy, France
[4] Ecole des Mines de Saint Etienne, France

March 23, 2007

Abstract

Engine torque estimation has important applications in the automotive industry: for example, automatically setting gears, optimizing engine performance, reducing emissions and designing drivelines.

A methodology is described for the on-line calculation of torque values from the gear, the accelerator pedal position and the engine rotational speed. It is based on the availability of input-torque experimental signals that are preprocessed (resampled, filtered and segmented) and then learned by a statistical machine-learning method.

Four methods, spanning the main learning principles, are reviewed in a unified framework and compared using the torque estimation problem: linear least squares, linear and non-linear neural networks and support vector machines. It is found that a non-linear model structure is necessary for accurate torque estimation. The most efficient torque model built is a non-linear neural network that achieves about 2% test normalized mean square error in nominal conditions.

Symbols and abbreviations

a_i, $a_{i,j}$, $\mathbf{a}$	parameters of functional forms to be learned
A_p	accelerator pedal position
c	correlation coefficient
C	regularization factor (SVMs)
C_p	clutch pedal position (mm)
d	distance covered by the vehicle (km)
$\dot{d}$	vehicle speed (km/h)
E_{emp}	empirical error
E_{reg}	regularized error
$f(t)$	true function dependency between the input signal and the engine torque signal
F_R	load at the rear axle (kg)
$g(t)$	estimate of $f(t)$, model function
ℓ	number of learning examples or data points used in empirical error
ℓ_V	number of validation data points
L	loss function
m	dimension of the input vector $\mathbf{u}$ including delays, $m = td_y + \sum_{i=1}^{N} td_i + N$
N	number of input signals used for the model
p	input signal power when calculating correlations, polynomial kernel order in SVM
r	gear engaged
$stop$	binary indication of stopped vehicle
S_1	early stopping parameter (maximal number of learning iterations where stopping error increases)
S_2	early stopping parameter (maximal number of learning iterations)
$td^{\max}$	maximum time delay
td_i	time delay for the i-th signal
td_y	time delay for the torque signal y
$\mathbf{u}$, $\mathbf{u}(t)$	input vector of the engine torque model (at time t)
$x_i(t)$	i-th signal at time t used as a component of $\mathbf{u}$
y	net engine torque ($N \cdot m$)
w_i	weights from the neurons to the output in neural networks
W	signal low-pass filter order
$\|\cdot\|_K^2$	functional norm in the RKHS defined by the kernel K
ε	resolution of the ε-insensitive norm in SVMs
λ	regularization weight factor
$\dot{\theta}$	engine speed ($rotation/min$)
IC	internal combustion (engine)
$nmse$	normalized mean square error
NN	neural networks
SVM	support vector machine
RKHS	reproducing kernel Hilbert space

1 Introduction

Engine torque estimation has been an active field of research in the last decade, primarily in relation to control apparatus that have become generalized in modern vehicles for optimizing engine performance, reducing traction wheel slip, reducing emissions and setting gears. Torque estimates are also needed for sizing gearboxes and other driveline components because, along with the temperature, they are a key factor in calculating mechanical damage.

Most previous research on torque estimation has relied on simple[1] internal combustion (IC) engine models. The inputs to these models typically include specific engine measures such as air or fuel mass flows (Chamaillard, et al. 2004, Karlsson & Fredriksson 1999, Scherer, et al. 1999, Dixon & Heslop 2000, Namba, et al. 1992), throttle angles (Hohmann, et al. 2000, Jankovic 2002, Scherer et al. 1999), manifold pressures (Sano 1995, Hohmann et al. 2000, Jankovic 2002, Scherer et al. 1999), sparks advances (Chamaillard et al. 2004, Karlsson & Fredriksson 1999) and crankshaft positions (Rizzoni, et al. 2003, Wang, et al. 1997).

The present work deals with estimating the net IC engine torque from readily available measures, namely accelerator pedal position, vehicle speed and engine rotational speed. Real torque measures, i.e. those obtained by customers in everyday driving conditions, interest car manufacturers because they enable a statistical description of the actual loads transmitted in the power trains. This work has been motivated by and applied to a car customer survey, in which 40 vehicles were equipped with a data acquisition system and lent to 40 customers for a month; during that time the specified signals were recorded in everyday driving conditions. The torque could not be directly recorded because in case of an accident the car manufacturer would need to be able to prove that the vehicle had not been significantly altered for the survey. This condition restricted the recorded signals to the ones considered here, from which the torque was estimated a posteriori.

This article presents a principled approach for creating torque models. based on statistical learning strategies that automatically build a model from experimental inputs (in this case torque sets). A statistical learning method encompasses a functional form and a way of tuning it to the data at hand: linear least squares (Ljung 1987), neural networks (Bishop 1995) and support vector machines (Vapnik 1998) will be considered here.

The article starts with a review of engine torque modelling, followed by a description and a first analysis of the experimental data available. Then, three important classes of statistical learning strategies that span most of the existing statistical learning principles—linear least squares, neural networks (NN) and support vector machines (SVM)—are introduced and compared for the torque estimation problem.

[1] Here "simple" means that the model evaluation is sufficiently rapid to allow real time applications.

2 Review of engine torque modelling

Many statistically founded methods for learning dynamic systems will be compared later in this article for an important practical problem, the estimation of engine torque. Car manufacturers need to know engine torque values in order to size power train components (such as gearboxes and crankshafts) and to regulate the engine and gearbox for optimized performance and reduced emissions.

Torque can be mechanically estimated from either of two viewpoints: the internal combustion engine and the car's longitudinal dynamics. As detailed in Appendix A, a car's longitudinal dynamics cannot properly predict the torque when external influences such as wind, wheel-on-road friction and slope are unknown. Since those factors are typically unknown, previous strategies for torque calculation have involved modelling the engine.

Models of internal combustion engine have various degrees of complexity. The most complex models are complete multi-physical descriptions of the combustion chamber (chemical, thermal and compressible gas analyzes) in which the torque can be obtained by integrating the pressure on the pistons (Heywood 1988, Vitek, et al. 2003). An example of a complex model is found in (Chalhoub, et al. 1999) where the effects of structural deformations on the engine friction torque are assessed. Such physically detailed models are computationally too demanding for control or pre-design.

Therefore most previous research on torque estimation has relied on simple models. In (Karlsson & Fredriksson 1999), for example, the validity of simple IC engine models for control is studied by comparing cylinder-by-cylinder and mean-cylinder models.

Simple torque models can be classified according to their inputs. Most are directly based on engine inputs such as the air and fuel mass flows and the ignition timing (Chamaillard et al. 2004, Karlsson & Fredriksson 1999, Namba et al. 1992). Equivalently, intake and exhaust pressure, engine speed and throttle angle have also been commonly used for torque estimation (Jankovic 2002, Hohmann et al. 2000, Scherer et al. 1999). Another engine internal variable, the difference in combustion chamber pressure, has been correlated to engine torque in (Sano 1995). Finally, crankshaft dynamics have been used to calculate engine torques in (Rizzoni et al. 2003, Wang et al. 1997).

Alternatively, simple torque models can be classified by their functional forms. In (Jankovic 2002) and (Hohmann et al. 2000), analytical relations based on elementary fluid dynamics yield the intake air flow (and thus the torque). More often, generic models are identified from time histories of measured engine variables: linear dynamic equations e.g. (Namba et al. 1992, Scherer et al. 1999), composition of polynomials (Chamaillard et al. 2004), composition of fuzzy linear dynamic equations (Maertens, et al. 2004), combination of polynomials (stochastic estimation of the coefficients), frequency analysis (Rizzoni et al. 2003) and neural networks (Hafner, et al. 2002).

Four statistical learning approaches for building simple torque models are com-

pared here. All four use as inputs measures available in any conventional vehicle[2]: the accelerator pedal's position, the car's speed and the engine rotation speed. The input set is restricted by the application motivating this research; more informative signals such as air and fuel flows and throttle position cannot be used for legal reasons in a survey of customers, as mentioned above.

Starting with such data and using no knowledge of the fuel injection controller, it is possible to build a physically based torque model that uses neither external car dynamics (since external friction sources are unknown) nor internal combustion relations (since fuel and air flows—as well as equivalent data such as throttle and crankshaft angles and manifold pressures—are unknown). This approach to torque reconstruction, which combines such inputs as accelerator pedal position (an internal command to the engine through the injection controller) and vehicle speed, is based on mixed internal and external dynamics.

3 Data description and analysis

3.1 Data description and preprocessing

The measured input signals that are considered here for estimating the net engine torque are

1. the distance covered by the vehicle, $d(t)$,
2. the vehicle speed, $\dot{d}(t)$,
3. the instantaneous load at the rear axle, $F_R(t)$,
4. a binary indication that the vehicle is stopped, $stop(t)$,
5. the clutch pedal position, $C_p(t)$,
6. the gear engaged, $r(t)$, with $r(t) = -2$ for gears not engaged, -1 for reverse, 0 for neutral, 1 to 5 for gears 1 to 5,
7. the engine rotation speed, $\dot{\theta}(t)$,
8. and the accelerator pedal position, $A_p(t)$.

The signal to predict is the engine torque $y(t)$. Many hours of measured input and output signals are available. The measured engine torque values were taken from the automatic gearbox calculator which defines a mapping to the torque value from the engine rotation speed, the air and fuel flows, various temperatures (cooling system, oil, outside), the accelerator pedal position and the gear engaged. The load at the rear axle was estimated by a sensor that measures displacements (in mm) between the chassis and the car body which are then converted to a load (in kg)

[2]Except for the load at the rear axle, F_R, which is studied here but is not useful in the best strategies.

	$d(t)$	$\dot{d}(t)$	stop	C_p	A_p	r	$\dot{\theta}$
F_R	-0.01	0.42	-0.43	0.17	-0.09	0.14	0.25
$d(t)$	-	-0.17	0.00	-0.15	-0.21	0.07	-0.25
$\dot{d}(t)$	-	-	-0.74	-0.04	0.34	0.41	0.77
stop	-	-	-	-0.20	-0.39	-0.14	-0.65
C_p	-	-	-	-	-0.14	-0.71	0.00
A_p	-	-	-	-	-	0.18	0.49
r	-	-	-	-	-	-	0.29

Table 1: Correlation coefficients, c, between input signals

by a vehicle-specific multiplier. Figure 1 shows examples of these signals; one can already see a similarity between the accelerator position and the torque. The signals, which were recorded independently at various frequencies, have been synchronized at 32 *Hz*. Downsampling and oversampling were performed by discarding excess samples and linear interpolation, respectively. Furthermore, the torque signal which was perturbed by a quantization noise has been processed by a low-pass mean filter of order $W = 20$. It is important to note that the filtering was useful because of the the quantization nature of the noise—not because of general experimental noises, which the statistical learning methods take into account. The filter order was chosen as a compromise between smoothing and preserving the signal extrema's amplitudes. Note also that the averaging filter introduces a delay of $(W - 1)/2$ periods, which was corrected on the filtered output.

Because the objective of this present work is to estimate the torque of a moving vehicle with a gear engaged, the *stop* and C_p signals are used to discard recordings when the vehicle is idle ($stop(t) = 1$) or the gear is not completely engaged ($C_p(t) > 0.1$). The brake pedal position is not needed for torque estimation; indeed, in terms of net engine torque, braking is treated simply as a case where no acceleration is applied.

3.2 Data analysis

Before turning to complex statistical learning methods, some intuition about the data is built by correlating input and output signals.

Firstly, linear correlation coefficients between input signals are calculated in Table 1. Some signals are slightly linearly correlated: the vehicle and the engine speeds, $\dot{d}(t)$ and $\dot{\theta}$, have $c = 0.77$, and the vehicle speed and the stop signal $stop(t)$ have $c = -0.75$. In both cases, the correlation is physically obvious. Similarly, the clutch pedal position C_p and the gear engaged r show some correlation ($c = -0.71$). The accelerator A_p and the engine rotation speed are also slightly correlated ($c = 0.49$).

Secondly, the correlation between each input and the unfiltered output (the torque, y) is studied. In order to investigate possible polynomial relationships, correlations between the input at the power p and the output are calculated in Table 2.

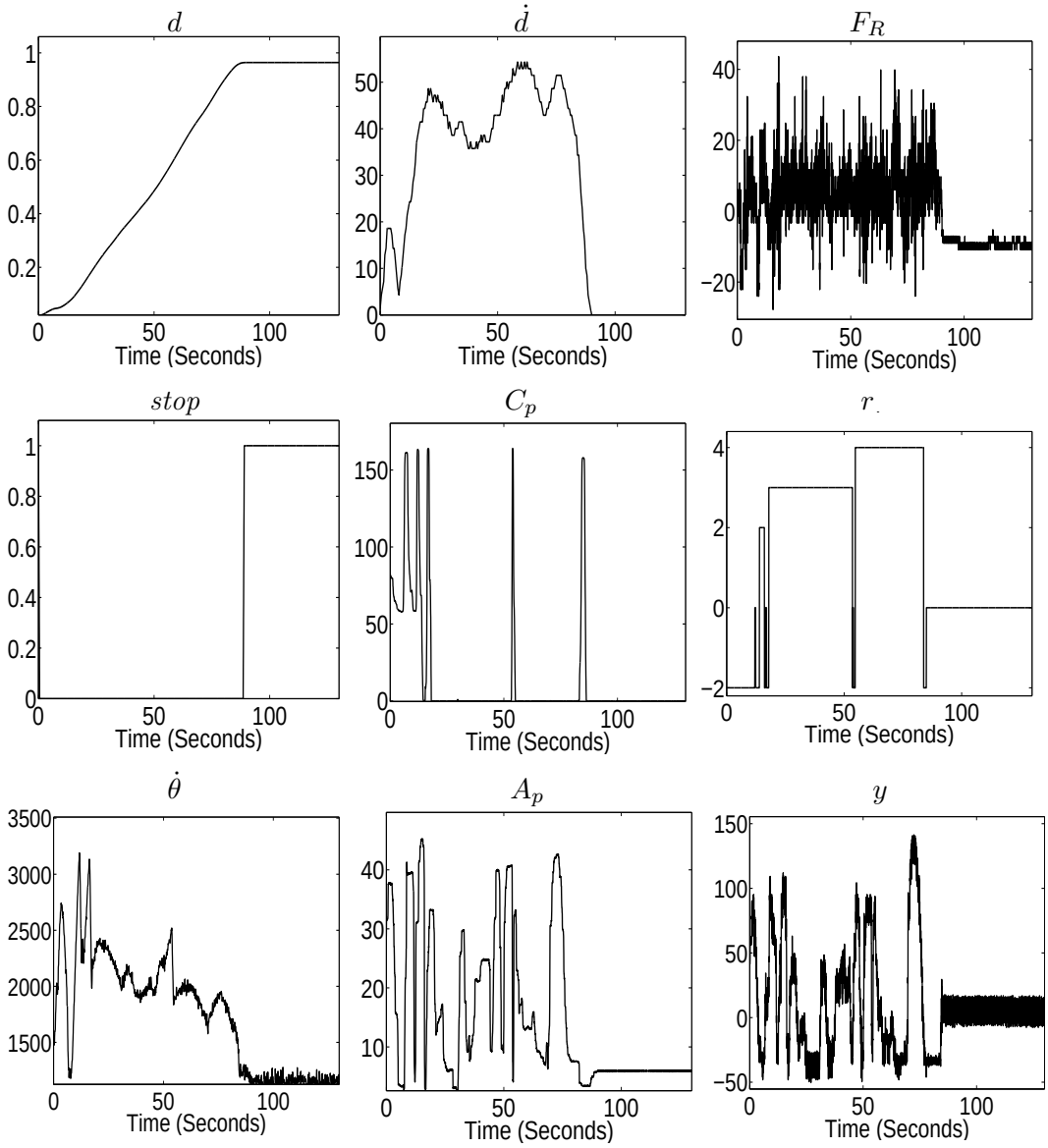

Figure 1: Examples of signals for a given data acquisition session. $d(t)$, distance covered by the vehicle, $\dot{d}(t)$, vehicle speed, $F_R(t)$, instantaneous load at the rear axle, $stop(t)$, binary indication of stopped vehicle, $C_p(t)$, clutch pedal position, $r(t)$, signal indicating which gear is engaged, $\dot{\theta}(t)$, engine speed, $A_p(t)$, accelerator pedal position, $y(t)$, net engine torque.

power p	F_R	d	$\dot{d}(t)$	$stop$	C_p	A_p	r	$\dot{\theta}$
1	-0.23	-0.21	0.17	-0.19	-0.13	0.91	0.12	0.30
2	-0.02	-0.23	0.12	-0.19	-0.19	0.95	0.25	0.30
3	-0.15	-0.24	0.09	-0.19	-0.19	0.92	0.11	0.28
4	-0.02	-0.23	0.06	-0.19	-0.19	0.87	0.21	0.26

Table 2: Correlation coefficients, c, between each input at the power p and the output torque, $y(t)$.

Figure 2 illustrates the relationships by plotting the output against each input. Table 2 and Figure 2 show that there is a strong relationship between y and the accelerator position, A_p, with a linear coefficient of correlation $c = 0.9$. Looking more closely at the figure, one sees that the relationship is more quadratic than linear for low torques. The table also supports this conclusion because the correlation between y and A_p^p is highest for $p = 2$. The engine speed, $\dot{\theta}$, is also slightly correlated to the torque ($c = 0.3$). The input signals other than these two have no significant polynomial correlation with the torque.

4 Statistical learning methods for engine torque modelling

The previous section has shown that acceleration and its square were highly correlated to the engine torque. In order to build a more accurate model of engine torque, the torque estimation problem will now be formulated to enable time delays and non-linear dependencies on inputs, in order to fit the general statistical learning framework.

The output (the torque, y) depends on N inputs x_i (e.g., A_p, $\dot{\theta}$, ...) and past values of the torque through

$$y(t) = f\left(\mathbf{u}(t)\right) + e(t) \tag{1}$$

where $e(t)$ represents the modelling error and the input vector $\mathbf{u}$ is

$$\mathbf{u}(t) = \left[x_1(t), \cdots, x_1(t - td_1), \cdots, x_N(t), \cdots, x_N(t - td_N), y(t-1), \cdots, y(t - td_y)\right] . \tag{2}$$

td_i is the maximal delay of the input signal x_i. It is assumed that the output signal value at a time t depends on its past values up to the time $t - td_y$. f is an unknown function of $m = td_y + \sum_{i=1}^{N} td_i + N$ variables.

Building a torque model, denoted g, now consists in approximating f from a finite set of measured input/output data, $(\mathbf{u}(t)/y(t))$. Such a regression problem, combining functional analysis and statistics, is the subject of statistical learning (Vapnik 1998). It is intuitive that g should reasonably fit the data, i.e. have a small

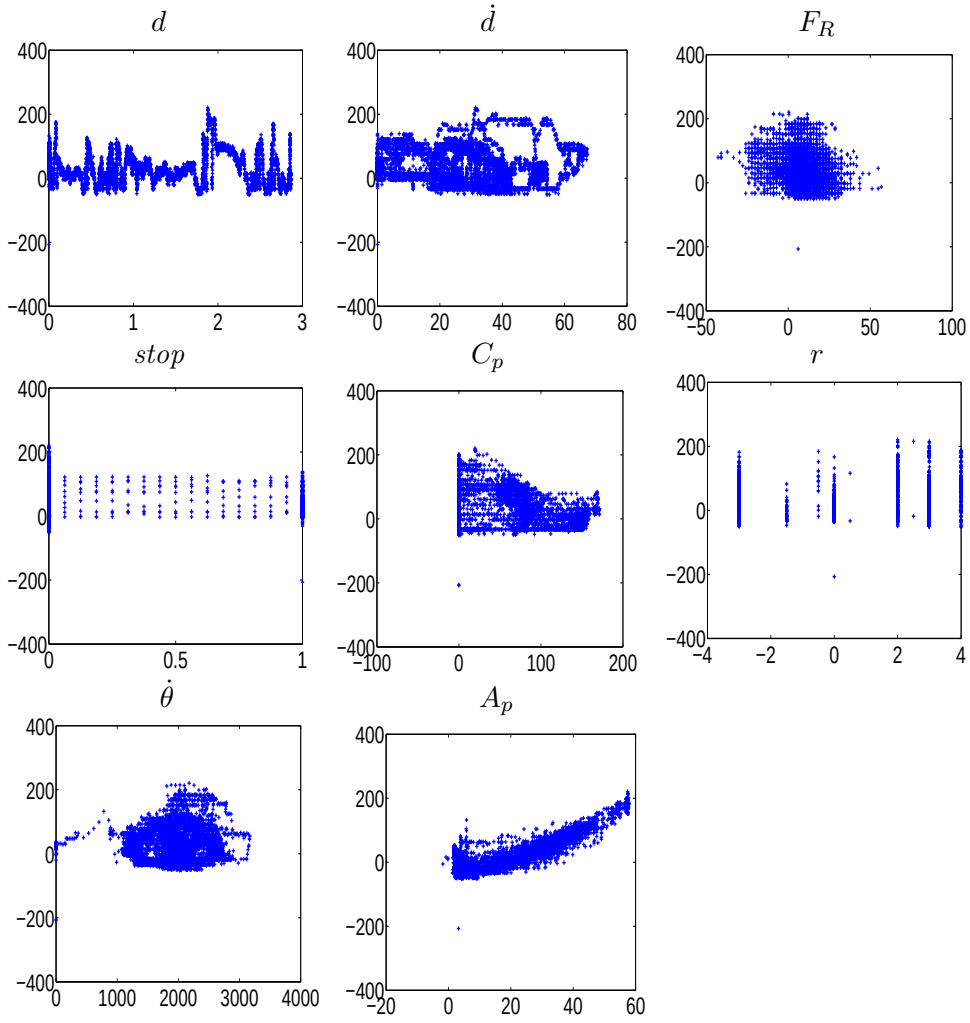

Figure 2: Net torque, $y(t)$, vs. each input signal. >From left to right and top to bottom: $d(t)$, $\dot{d}(t)$, $F_R(t)$,$stop(t)$, $C_p(t)$, $r(t)$, $\dot{\theta}(t)$, $A_p(t)$.

empirical error

$$E_{emp}(g) = \frac{1}{\ell}\sum_{i=1}^{\ell} L(y(t_i), g(\mathbf{u}(t_i))) \tag{3}$$

where ℓ is the number of samples and L is a *loss function*, e.g. $||y(t_i) - g(\mathbf{u}(t_i))||^2$ or $|y(t_i) - g(\mathbf{u}(t_i))|_\varepsilon$ as explained later. However, blindly minimizing E_{emp} over a large class of functions[3] $\mathcal{G}$ would make g exactly fit the data but with a poor prediction ability because *i)* g would learn data noise, and *ii)* g would not uniquely be

[3]Examples of "large" functional spaces are polynomials of a sufficiently high degree or finite trigonometric sums. Other examples are the functional forms spanned by neural networks or support vector machines, as seen later in the text.

determined by the data or, in mathematical terms, $\min_{g \in \mathcal{G}} E_{emp}$ would be ill-posed. Statistical learning theory is therefore concerned with constraining the minimization of the empirical error to an appropriately small functional space; this is generically known as regularization (Evgeniou, et al. 2000). Most statistical learning strategies implement it by minimizing a *regularized error*, which is the sum of the empirical error and a *functional regularization* term

$$E_{reg}(g) = E_{emp}(g) + \lambda ||g||_K^2 \ . \tag{4}$$

The functional regularization enforces some smoothness and flatness on the function; λ is a parameter that controls the trade-off between fitting the data and having a regular function. It is a functional norm or, more precisely, the norm in the reproducing kernel Hilbert space (RKHS) defined by the positive definite function (kernel) K (see (Evgeniou et al. 2000) for further explanation). An example of $||g||_K^2$ when g is a finite dimensional linear function as in Equation (5) is the L_2 norm of its coefficients vector, $\|g\|_K^2 = \sum_{i=1}^m a_i^2$ (see also footnote 4 and Section 4.3 on support vector machines).

Besides functional regularization, there exist *computational regularization* techniques, the most general example of which is *cross-validation* (Efron 1983). This technique aims to estimate the prediction error by dividing the data into subsets and repeatedly minimizing E_{reg} while leaving out one subset. These subsets are used at each cross-validation iteration to calculate one prediction error occurrence. Averaging these prediction errors yields the estimate. Cross-validation is used here to choose *model hyper-parameters*, which are fixed when minimizing E_{reg}, such as λ in Equation (4). Two other computational regularization techniques, early stopping and weight decay, will be presented with neural networks.

In brief, a statistical learning method is defined by a functional space $\mathcal{G}$ in which the regularized error E_{reg} is minimized, a loss function L, a functional regularization $||g||_K^2$ and optionally a strategy for computational regularization. These items will be described next for the four torque estimation methods (linear least squares, linear and non-linear neural networks and support vector machines). The following assumes that the reader is familiar with optimization concepts, which can be found in (Minoux 1986).

4.1 Linear Least Squares modelling

This is the baseline method for regression. It is computationally and conceptually the simplest approach and may satisfactorily predict the torque since it is highly correlated to the acceleration. The salient feature of linear models is their intrinsic stiffness which makes them less prone to data overfitting; this saves the effort required for regularization, but also limits their accuracy in non-linear cases. The functional form of the estimated output is

$$g(\mathbf{u}(t)) = \sum_{i=1}^{m} a_i u_i(t) + a_0 \tag{5}$$

where $u_i(t)$ is the i-th component of the input vector $\mathbf{u}(t)$.

The loss function is a quadratic norm and there is no regularization. Thus, the coefficients $a_0, \cdots, a_m$ minimize the empirical error

$$E_{emp}(g) = \frac{1}{\ell} \sum_{j=1}^{\ell} \left(y(t_j) - \sum_{i=1}^{m} a_i u_i(t_j) - a_0 \right)^2 \tag{6}$$

which gives, in matricial notation,

$$\min_{\mathbf{a}} \|\mathbf{y} - U\mathbf{a}\|^2 \tag{7}$$

where $\mathbf{a} = [a_0, \cdots a_m]^t$, $\mathbf{y} = [y(t_1), \cdots, y(t_\ell)]$ and

$$U = \begin{pmatrix} 1 & u_1(t_1) & \cdots & u_m(t_1) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & u_1(t_\ell) & \cdots & u_m(t_\ell) \end{pmatrix} .$$

The solution of this least squares minimization problem is given by the normal equations,

$$U^t U \mathbf{a} = U^t \mathbf{y} \ . \tag{8}$$

4.2 Neural Networks

Neural networks (NNs) are a very popular statistical learning method (Bishop 1995, Ripley 1996). The functional form of a NN is

$$g(\mathbf{u}(t)) = \sum_{j=1}^{N_h} w_j \Phi_j \left(b_j + \sum_{i=1}^{m} a_{i,j} u_i(t) \right) + a_0 \tag{9}$$

which is metaphorically described as the linear combination of N_h "hidden layer neurons" Φ_j's. In *non-linear NNs*, the Φ_j's are sigmoidal functions such as the hyperbolic tangent. In this work, *linear NNs* are additionally considered where the Φ_j's are identity functions. Non-linear NNs have three interesting properties, *universal approximation*, *parsimony* and *complexity control*. Universal approximation means that they can approximate arbitrarily well any function providing N_h is large enough (a property shared by other function classes such as polynomials and Fourier series). They are parsimonious because their number of internal parameters ($1 + mN_h + 2N_h$ for a_0, the $a_{i,j}$'s the b_j's and w_j's) grows linearly with the dimension of the input space, m. Complexity control means that by changing N_h one can control the variance of the function independently from other factors such as the number of data or the input space dimension.

A NN learns the data by solving the non-linear least-squares problem

$$\min_{\mathbf{w},\mathbf{a}} \frac{1}{\ell} \sum_{i=1}^{\ell} (y(t) - g(\mathbf{u}(t_i)))^2 + \lambda \left(\sum_j w_j^2 + \sum_{i,j} a_{i,j}^2 \right) \tag{10}$$

The Levenberg-Marquardt optimization algorithm is well suited to solving this problem (Levenberg 1944, Marquardt 1963). It is a gradient-based optimizer that, for non-linear least-squares problems, is a quasi-Newton method (Le Riche & Guyon 1999). Evaluation of the gradient of the objective function (Equation (10)) with respect to the parameters (or weights) $\mathbf{w}$ and $\mathbf{a}$ is performed by a computationally efficient implementation of chain rule differentiation, a technique known as "error back-propagation" in the field of NNs (Bishop 1995).

The last penalty term in Equation (10), referred to as *weight decay*, is, rigorously speaking, a computational regularization strategy, although it seems closely related to functional regularization[4] (see Equation (4) and (Canu & Elisseff 1999, Girosi et al. 1995)).

Another computational regularization strategy, *early stopping*, is used for learning NNs. Both versions of this strategy involve stopping the minimization of Equation(10) before convergence.

In one version, a "stopping" subset of the learning data set is isolated and used exclusively to control the empirical error—it is not involved in defining Levenberg-Marquardt search directions. Whenever the empirical error on the stopping subset increases for S_1 Levenberg-Marquardt iterations, optimization is stopped and the solution found S_1 iterations earlier is chosen even though the empirical error on the learning subset might have kept decreasing. This strategy is based on the supposition that if the stopping error increases while the learning error decreases, optimization is probably fitting the noise of the learning data.

In the other version, the search is stopped after a given number of Levenberg-Marquardt iterations, S_2.

In this work, both stopping criteria are used and optimization is stopped whenever either is met. The NN regularization hyper-parameters λ and S_2 are determined through cross-validation.

Finally, note that the difference between linear least squares and linear NN is that the latter is subject to the specified computational regularization strategies (weight decay and early stopping). Otherwise, these two models share the same functional form and empirical error.

4.3 Support Vector Machines

Support Vector Machines (SVM) (Schölkopf & Smola 2001, Vapnik 1998) in regression minimize the regularized error E_{reg} of Equation (4) where the loss function L of the empirical error is the *ε-insensitive loss function*

$$E_{emp}(g) \;=\; \frac{1}{\ell}\sum_{i=1}^{\ell} |y(t_i) - g(\mathbf{u}(t_i))|_{\varepsilon} \tag{11}$$

[4] In the case of a linear NN, weight decay is a functional norm in a RKHS of (finite) dimension $\mathbb{R}^{n+1}$, whose basis vectors are $\begin{array}{ccc} e_0 : \mathbb{R}^n & \to & \mathbb{R} \\ x & \to & 1 \end{array}$, $\begin{array}{ccc} e_i : \mathbb{R}^n & \to & \mathbb{R} \\ x & \to & x_i \end{array}$ and whose kernel is $K(x,y) = \sum_{i=1}^{n} x_i y_i + 1$. The link between weight decay and functional norms is still a subject for research in the case of non-linear NNs (Canu & Elisseff 1999, Girosi, et al. 1995).

and $|\cdot|_\varepsilon$ means

$$|x|_\varepsilon = \begin{cases} 0 & \text{if } |x| < \varepsilon \\ |x| - \varepsilon & \text{otherwise.} \end{cases} \tag{12}$$

With the ε-insensitive loss function, any output which is less than ε away from the data is treated as a perfect interpolant. ε can be interpreted as the resolution of the measures; it also controls the regularity of the function that minimizes E_{reg} because increasing it reduces the emphasis on interpolating the data and increases the degrees of freedom available for decreasing the regularization term of Equation (4).

Note that the functional norms $\|g\|_K^2$ will not be explicit in the framework of classical SVMs because the first order optimality conditions (the Karush Kuhn and Tucker conditions) will be used to transform them into linear combinations of K.

However, the choice of the kernel K defines the reproducing kernel Hilbert space (RKHS) in which a minimizer of E_{reg} (Equations (4) and (11)) is sought. In other words, the kernel determines the functional form of the solution which can be written (as any function of the RKHS)

$$g(\mathbf{u}(t)) = \sum_{i=1}^{\ell} a_i K(\mathbf{u}(t_i), \mathbf{u}(t)) + a_0 \ . \tag{13}$$

Because the ε-insensitive loss function is not differentiable, slack variables ξ and ξ^* are introduced into the original SVM problem; this yields an equivalent quadratic optimization problem:

$$\begin{array}{l} \min_{g,\xi,\xi^*} \Phi(g,\xi,\xi^*) = \frac{C}{\ell}\sum_{i=1}^{\ell}(\xi_i + \xi_i^*) + \frac{1}{2}\|g\|_K^2 \\ \\ \text{subject to the constraints:} \\ g(\mathbf{u}(t_i)) - y_i \le \varepsilon + \xi_i \quad \forall i \in [1, \cdots \ell] \\ y_i - g(\mathbf{u}(t_i)) \le \varepsilon + \xi_i^* \quad \forall i \in [1, \cdots \ell] \\ \xi_i, \xi_i^* \ge 0 \quad \forall i \in [1, \cdots \ell] \end{array} \tag{14}$$

where $C = \frac{1}{2\lambda}$. This minimization problem is convex, so it is equivalent to solve its dual form where the Lagrange multipliers associated with the ε-tube constraints are introduced (denoted α_i and α_i^*). Making use of the Karush-Kuhn and Tucker conditions in the dual form to eliminate the slack variables, the SVM regression problem turns out to be the following quadratic programming problem in α_i and α_i^* (Schölkopf & Smola 2001, Smola & Schölkopf 1998, Vapnik 1998):

$$\begin{array}{l} \min_{\alpha_i,\alpha_i^*} \varepsilon\sum_{i=1}^{\ell}(\alpha_i^* + \alpha_i) - \sum_{i=1}^{\ell} y_i(\alpha_i^* - \alpha_i) + \frac{1}{2}\sum_{i,j=1}^{\ell}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)K(\mathbf{u}(t_i), \mathbf{u}(t_j)) \\ \\ \text{subject to the constraints:} \\ \sum_{i=1}^{\ell}(\alpha_i^* - \alpha_i) = 0 \\ 0 \le \alpha_i, \alpha_i^* \le \frac{C}{\ell} \quad \forall i \in [1, \cdots \ell] \end{array} \tag{15}$$

a_0 is the Lagrange multiplier associated to the constraint $\sum_{i=1}^{\ell}(\alpha_i^* - \alpha_i) = 0$. The solution of the regression problem (the minimizer of E_{reg}) is finally

$$g(\mathbf{u}) = \sum_{i=1}^{\ell}(\alpha_i - \alpha_i^*)K(\mathbf{u}(t_i), \mathbf{u}) + a_0 \ . \tag{16}$$

Note that the data points $\mathbf{u}(t_i)$ for which the error is less than ε have their α_i and α_i^* equal to 0 since they are Lagrange multipliers associated to inactive constraints. Furthermore, α_i and α_i^* cannot be simultaneously equal to 0, so only a few points, called the "support vectors", have either α_i or α_i^* different from zero. The support vectors are the only data points that determine the model g (Equation (16))—the other points could be removed without changing the SVM; this property is known as *sparsity*.

In SVMs, two hyper-parameters are tuned by cross-validation, C $(= 1/(2\lambda))$ and ε.[5] Polynomial kernels will be used in this work,

$$K(u, u') \ = \ (< u, u' > +1)^p \ , \tag{17}$$

where $< u, u' >$ denotes the scalar product between u and u'.

This presentation of the statistical learning methods under consideration for torque estimation will close with a comparative review of their theoretical features. One should be careful here not to rush to conclusions because, as the No Free Lunch Theorem in optimization (Wolpert & MacReady 1995) states, there is no such a thing as a universally better performing method. For this reason the numerical tests performed later on the torque estimation problem remain necessary.

For certain choices of kernels such as Gaussian radial basis functions, it has been shown (Dyn 1987) that SVMs, like NNs, are universal approximators. However, the universal approximation property is not of much practical benefit in itself since it should be balanced against the associated risk of poor prediction. More important in practice is the difficulty of the optimization problem that needs to be solved to learn the methods. Learning NNs is complex because it involves a non-linear least squares problem where the Levenberg-Marquardt algorithm may converge to a local minimum. On the other hand, the linear least squares method can be tuned to the data by a simple linear system resolution. Learning SVMs involves solving a classical quadratic programming problem where convergence to a global optimum is guaranteed. With regard to computer memory usage, SVMs typically require more memory than NNs, as shown in Problem (15) where there are ℓ^2 terms $K(\mathbf{u}(t_i), \mathbf{u}(t_j))$. To save some memory, SVM implementations can make use of the sparsity property (Collobert & Bengio 2001): Problem (15) is solved as a series of smaller quadratic programming problems, each of which corrects a guess of the support vectors. In

[5] Cross-validation on ε is possible because the criterion that will be used to compare all learning methods in Section 5 is the normalized mean square error—whose definition does not depend on ε.

Linear Least Squares	
functional space	$g(x) = \sum_{i=1}^{m} a_i x_i + a_0$
loss function L	least squares
functional regularization	none
computat. regularization	none
optimization pb.	linear least squares

Neural Networks	
functional space	$g(x) = \sum_{j=1}^{N_h} w_j \Phi_j \left(\sum_{i=1}^{m} a_{i,j} x_i\right) + a_0$, $\Phi_j(x) = \tanh(x)$ or x
loss function L	least squares
functional regularization	none
computat. regularization	weight decay, early stopping, cross-validation (λ and S_2)
optimization pb.	non-linear least squares (Levenberg-Marquardt)

Support Vector Machines	
functional space	$g(x) = \sum_{i=1}^{\ell} a_i K(u(t_i), x) + a_0$
loss function L	ε-insensitive
functional regularization	$1/(2C)\|g\|_K^2$
computat. regularization	cross-validation (C and ε)
optimization pb.	constrained quadratic programming

Table 3: Summary of the features of the statistical learning methods considered.

each subproblem, many α_i and α_i^* are set to 0 (non support points) and not considered in the optimization. This guess is iteratively updated until a solution to the quadratic problem (15) is found.

All the characteristics of the methods presented are summarized in Table 3.

5 Experimental results

5.1 Experimental setup

The four statistical learning methods previously described are now compared for estimating an IC engine net torque.

The relationship between the engine torque, the accelerator pedal position $A_p(t)$ and the engine speed $\dot{\theta}$ depends on which gear is engaged, as measured by the discrete signal $r(t)$. Here we have adopted the simplest way to cope with this dependency: building a separate model for each gear. This can be seen as a rough way of introducing expert knowledge into the models. Other approaches would imply making $r(t)$ one of the inputs u_i in Equation (2) with the drawback of increasing the input space dimension and the complexity of the learning task.

A large number of experimental data, representing several hours of recording, are available here. They translate into a number of $(\mathbf{u}, y)$ pairs that vary with the maximum delay, $td^{\max} = \max_{i=y,1\ldots N}(td_i)$, because the input signals of relation (2) have to start $td^{\max}$ time steps after the beginning of usable sequences. Short sequences

may also disappear from the inputs as their length becomes smaller than $td^{\max}$. On average, there are on the order of 140000 data points for each gear.

The data are divided into learning, validation and test sets. The *learning set* consists in general of the ℓ points used to calculate the empirical error (Equation (3)) when tuning the models parameters (the a and w parameters). An exception is made for early stopping, when part of the learning set is put aside to control learning as explained in Section 4.2. The *validation set* is composed of the ℓ_V points used to evaluate each tuned model's performance and cross-validate the hyper-parameters (see Section 4). Performance is quantified in terms of the *normalized mean square error*

$$nmse = \frac{\sum_{i=1}^{\ell_V}(y(i) - g(u(t_i)))^2}{\sum_{i=1}^{\ell_V} y(i)^2} \ . \tag{18}$$

The prediction (or *generalization*) ability of a model is evaluated by its *nmse* on the *test set*. The data are correlated in time, so care is taken to split them so that contiguous sequences of signals are allocated to the sets. For each gear, the learning and validation set sizes are $\ell = \ell_V = 20000$, the early stopping set is made up of 30% of the learning set and the test sets have about 100000 points.

5.2 Setting hyper-parameters

A first experiment is performed in order to choose the value of the time delays td_i and td_y. These should be chosen in accordance with the driveline dynamics. The time delays are taken to be the same for all input signals but the torque, and are varied from 0 to 40 for a non-linear NN composed of 5 tanh neurons in the hidden layer and having A_p and $\dot{\theta}$ as inputs. In all cases, $td_y = 0$ in the best performing variants: so there is no recursivity in the statistical models. The results in terms of *nmse* are shown in Figure 3 for gears 2 and 5. Optimal delays are 30 for gear 2 and 40 (or more) for gear 5. The smaller delays for gear 2 than gear 5 reflect faster dynamics. It is striking however that the improvement in *nmse* is no longer significant beyond delays of 20. Therefore, in accordance with Occam's Razor[6], time delays of 20 are chosen for a saving of $10 \times (N + 1)$ inputs.

In the case of SVMs, the results presented here correspond to third order polynomials ($p = 3$ in Equation 17), which were found to have the best performance from a range of $p = 1$ to 4.

The hyper-parameters of NNs and SVMs are chosen by cross-validation. For NNs, these parameters are the weight decay factor, λ, and the number of optimization iterations (early stopping), S_2. They are selected from the sets of possible values $\{0.01, 0.1, 1\}$ and $\{100, 500, 1000\}$, respectively. The other early stopping

[6]Named after the 14th century logician and Franciscan friar, William of Occam; the general scientific meaning of Occam's razor is this: if you have two equally likely solutions to a problem, pick the simplest. In statistical learning, Occam's Razor corresponds to the principle of structural risk minization (Vapnik 1998) which, out of two models of similar observed performance, advocates selecting the simplest.

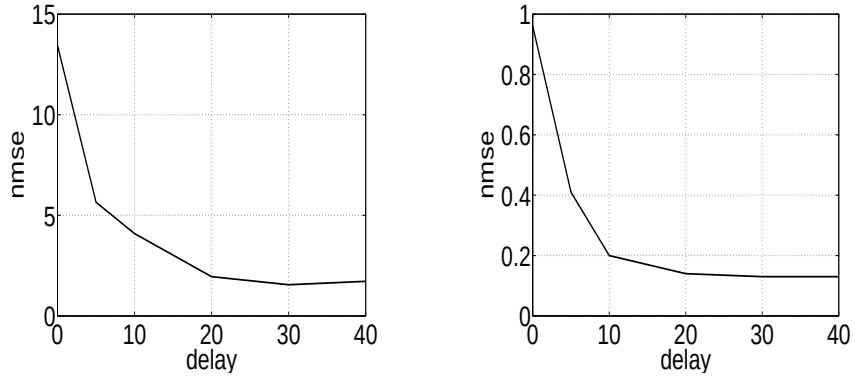

Figure 3: Validation normalized mean square error vs. time delays. The model is a neural network with 5 tanh neurons in the hidden layer and accepting A_p and $\dot{\theta}$ as inputs. (left) Gear 2. (right) Gear 5.

Model : hyper-parameters	Input Signals			
	A_p	$A_p, \dot{\theta}$	$A_p, \dot{\theta}, \dot{d}$	$A_p, \dot{\theta}, \dot{d}, F_R$
NN, 5 tanh(x)'s, gear 2 : (S_2, λ)	(500,0.1)	(500,0.01)	(1000,0.01)	(500,0.1)
NN, 5 tanh(x)'s, gear 5 : (S_2, λ)	(100,1)	(500,0.1)	(500,0.1)	(1000,0.1)
3rd order polynomial SVM, gear 2 : (C, ε)	(10,0.05)	(10,0.05)	(10,0.05)	-
3rd order polynomial SVM, gear 5 : (C, ε)	(10,0.05)	(10,0.05)	(100,0.05)	-

Table 4: Optimal values of hyper-parameters obtained by cross-validation, filtered torque.

parameter, S_1, is set equal to 10. Table 4 summarizes the optimal values of the hyperparameters. The optimal settings of the NNs change with respect to the network input set, but it is clear that more regular functions (lower S_2 and higher λ) perform better on the higher gears, which indeed have slower dynamics. For SVMs, the two hyper-parameters are the regularization factor, C, and the width of the ε-insensitive norm, ε, which are taken from the sets {10, 100, 1000} and {0.05, 0.1, 0.15}, respectively. The optimal values of these hyper-parameters, given in Table 4, are constant at $C = 10$, $\varepsilon = 0.05$.

5.3 Comparison of models

The statistical learning methods (linear least squares, NNs and SVMs) are now compared for six sets of input signals: $\{A_p\}$, $\{A_p, \dot{\theta}\}$, $\{A_p, \dot{\theta}, \dot{d}\}$, $\{A_p, \dot{\theta}, F_R\}$, $\{{A_p}^2, \dot{\theta}\}$ and $\{\dot{\theta}, \dot{d}, F_R\}$. Tables 5 and 6 show the validation *nmse* obtained for the different models. The following conclusions can be made:

- Averaged over both gears, the best[7] model is a non-linear NN with 5 tanh neurons in the hidden layer accepting as inputs ${A_p}^2$ and $\dot{\theta}$. It accurately estimates the torque with a validation *nmse* of 1.9% on gear 2 and 0.14% on gear 5. The faster dynamics of gear 2 make it more difficult to learn than gear 5. Squaring the accelerator position is not critical to this NN model because the model is non-linear, and using A_p and $\dot{\theta}$ as inputs yields a similar prediction performance.

- Using the accelerator position as input is critical to the method, as illustrated by the major deterioration in *nmse* seen in the rightmost columns of Tables 5 and 6.

- For gear 2, the dependency of engine torque on accelerator position is clearly non-linear since using non-linear models reduces the *nmse* by about 3%. The non-linearity is, as the correlation coefficients of Table 2 suggest, partly represented by the square of the accelerator position, since linear models in gear 2 have their *nmse* decrease by 2.3% with a substitution of A_p for ${A_p}^2$. This non-linearity fades away for the fifth gear, where non-linear models are only 0.6% better than linear ones.

- The best input signal set (of the sets tried here) is $\{A_p, \dot{\theta}\}$ (or equivalently $\{{A_p}^2, \dot{\theta}\}$) for both gears. On average over all models, changing the input from $\{A_p\}$ to $\{A_p, \dot{\theta}\}$ improves the torque estimates by about 3%. Adding $\dot{d}$ to $\{A_p, \dot{\theta}\}$ does not produce further significant progress. Such a result was partly expected since $\dot{\theta}$ is equal to $\dot{d}$ multiplied by a gear-dependent factor. Nevertheless, redundant signals may be useful to neutralize measuring noise; this explains the marginal improvement seen for gear 2, but this progress is so small that Occam's razor can be invoked to remove $\dot{d}$ from the inputs. Finally, adding a rear axle load estimate, F_R, to the inputs does not improve the torque estimation.

Figures 4 and 5 illustrate the performance of the best model for gears 2 and 5. Figure 4 plots the real versus the estimated torques, and shows again that the model is more accurate for gear 5 than for gear 2 (points are more concentrated around the $y = x$ axis in the first plot than in the second one). For both gears, it is evident that most points departing from the optimal $x = y$ region are above it at low torque values; this indicates a tendency to underestimate low torques. Figure 5 is a time representation of the estimated and real torque signals. While the overall dynamics of the engine is accurately reproduced by the best NN model, the most important errors occur for gear 2 in unstable regions (for example around 260 *sec.*).

To sum up performance on the validation set, the best model found using A_p, $\dot{\theta}$, $\dot{d}$ and F_R as possible inputs is a non-linear neural network composed of five

[7] More precisely, the best model of the ones studied in this article. In particular, it is likely that more informative inputs for torque—such as fuel and air flows, spark advances and throttle angles—would yield more robust estimations.

Models	Input Signals					
	A_p	$A_p, \dot{\theta}$	$A_p, \dot{\theta}, \dot{d}$	$A_p, \dot{\theta}, \dot{d}, F_R$	${A_p}^2, \dot{\theta}$	$\dot{\theta}, \dot{d}, F_R$
Linear Model	9.31	6.43	6.29	6.35	3.94	14.15
Linear NN	9.26	6.26	6.07	6.19	3.89	16.84
NL NN, 4 $\tanh(x)$ & 1 x	6.35	1.96	1.91	1.91	2.22	12.74
NL NN, 5 $\tanh(x)$'s	6.30	1.95	1.91	1.91	1.90	12.25
3rd order polyn. SVM	6.29	2.47	2.12	-	-	-

Table 5: Comparison of models validation $nmse$, gear 2, filtered torque.

Models	Input Signals					
	A_p	$A_p, \dot{\theta}$	$A_p, \dot{\theta}, \dot{d}$	$A_p, \dot{\theta}, \dot{d}, F_R$	${A_p}^2, \dot{\theta}$	$\dot{\theta}, \dot{d}, F_R$
Linear Model	2.49	1.70	1.78	1.85	5.40	12.41
Linear NN	1.86	1.21	1.30	2.85	4.59	14.22
NL NN, 4 $\tanh(x)$ & 1 x	1.28	0.14	0.17	0.23	0.12	14.18
NL NN, 5 $\tanh(x)$	1.28	0.14	0.16	0.21	0.14	26.00
3rd order polyn. SVM	1.74	0.28	0.73	-	-	-

Table 6: Comparison of models validation $nmse$, gear 5, filtered torque.

tanh neurons in the hidden layer with A_p and $\dot{\theta}$ as selected inputs. The prediction performance of this best model will now be evaluated on a test set composed of data that have never been used before, corresponding to experiments spanning a 1 month period, and divided into 6 subsets.

Table 7 gives the test $nmse$ results for the two gears. The model performs globally well ($nmse < 4\%$ for gear 2 and $nmse < 2\%$ for gear 5) except for test set 2. Close examination of test set 2 shows that most badly predicted data are associated with experiments run on the same day. Removing that day's data from set 2 reduces the $nmse$ values to 1.5% and 0.16% for gears 2 and 5, respectively. It is therefore likely that the data gathered on that day do not represent nominal driving conditions. It is noteworthy that detecting atypical data can be an interesting side application of statistical learning.

6 Concluding remarks

This article has shown that IC engine torque values can be accurately estimated on-line, in nominal conditions, using statistical learning methods from signals available in any conventional vehicle: the gear engaged (r), the accelerator pedal position (A_p) and the engine rotational speed ($\dot{\theta}$).

A wide spectrum of statistical methods for building the torque model (linear least

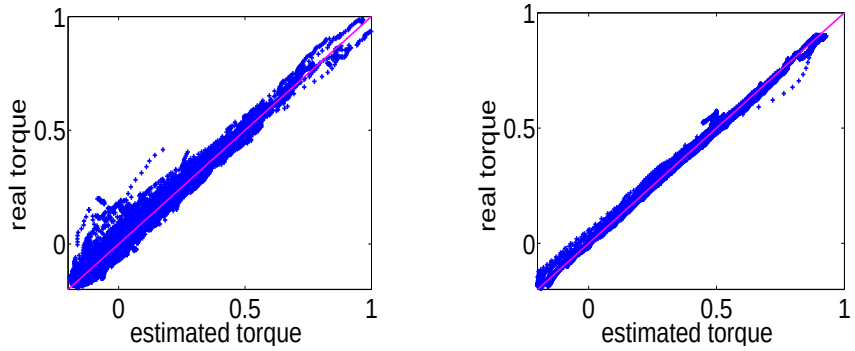

Figure 4: Examples of real vs. estimated engine torque on validation set, best model (NN with 5 tanh neurons using $A_p(t)$ and $\dot{\theta}(t)$ as inputs). (left) Gear 2. (right) Gear 5.

Test sets	size	*nmse* gear 2	size	*nmse* gear 5
Set 1	4460	0.71	16455	0.24
Set 2	8014	11.27	26278	4.18
Set 3	4531	2.55	107510	1.88
Set 4	15324	3.81	53739	0.35
Set 5	7693	2.06	18902	0.20
Set 6	8463	2.06	35600	0.17

Table 7: Normalized mean-square error on test sets using the best neural network model.

squares, linear and non-linear neural networks and support vector machines) have been compared. They differ in their functional forms, in how they are compared to the data (quadratic or ε-insensitive norms), and in how they are regularized. Non-linear neural networks performed the best with a test normalized mean square error of the order of 2%. The performance of support vector machines was close to that of the non-linear neural networks. Linear models were unable to predict torque correctly, especially at low gears.

The approach's main limitation is that experimental values of r, A_p, $\dot{\theta}$ and the torque must be available for each type of vehicle considered. Torque estimates deteriorate profoundly in the absence of A_p as an input signal.

Another limitation is that the method—at least in the form presented here—is not robust with regard to such changes of working conditions as major variations in ambient pressure or temperature because they affect the relationship between A_p and the torque.[8] The methodology presented here could be applied to developing a

[8]For example, at a high ambient temperature the spark is retarded to avoid engine knock.

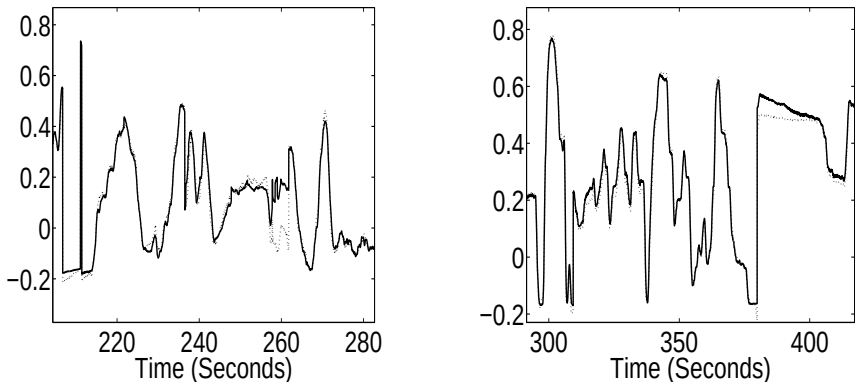

Figure 5: Examples of real (solid line) and estimated (dashed line) engine torques on validation set, best model (NN with 5 tanh neurons using $A_p(t)$ and $\dot{\theta}(t)$ as inputs). (left) Gear 2. (right) Gear 5.

more robust estimation system, but complementary inputs such as air, fuel or coolant temperatures would be required.

More generally, the statistical learning approaches summarized and illustrated here for torque estimation provide a way of diversifying the sensors data processings. In the near future, such redundant information may be exploited to detect abnormal use conditions and sensor failures.

Acknowledgement: The authors would like to thank the reviewers for their pertinent comments on the first version of this article. We also thank Adam Funk for his help in editing the article.

References

C. Bishop (1995). *Neural Networks for Pattern Recognition*. Oxford Univ. Press.

S. Canu & A. Elisseff (1999). 'Why sigmoid-shaped functions are good for learning'. In *Snowbird 99 workshop*, Utah, US.

N. G. Chalhoub, et al. (1999). 'Effects of structural deformations of the crank-slider mechanism on the estimation of the instantaneous engine friction torque.'. *Journal of Sound and Vibration* **224**(3):489–503.

Y. Chamaillard, et al. (2004). 'A simple method for robust control design, application on a non-linear and delayed system: engine torque control'. *Control Engineering Practice* **12**(4):417–429.

R. Collobert & S. Bengio (2001). 'SVMTorch: Support Vector Machines for Large-Scale Regression Problems'. *Journal of Machine Learning Research* **1**:143–160.

R. D. E., et al. (1986). 'Learning internal representations by error propagation'. In R. D. E., M. J. L., & P. R. Group (eds.), *Parallel Distributed Processing : Explorations in the Microstructure of Cognition, Foundations*, vol. 1, chap. 8. MIT Press, Cambridge, MA.

J. Dixon & G. N. Heslop (2000). 'Engine torque control'. Patent US 006035252A.

N. Dyn (1987). 'Interpolation of scattered data by radial functions'. In C. Chui, L. Schumaker, & F. Utreras (eds.), *Topics in multivariate approximation*. Academic Press, New York.

B. Efron (1983). 'Estimating the error rate of a prediction rule: Improvement on cross-validation'. *J. of the American Statistical Association* **78**:316–331.

T. Evgeniou, et al. (2000). 'Regularization Networks and Support Vector Machines'. *Advances in Computational Mathematics* **13**(1):1–50.

F. Girosi, et al. (1995). 'Regularization theory and neural networks architectures'. *Neural Computation* **7**:219–269.

M. Hafner, et al. (2002). 'Model-based control design for IC engines on dynamometers: the toolbox OPTIMOT'. In *15th IFAC congress*, Barcelona, Spain.

J. B. Heywood (1988). *Internal Combustion Engine Fundamentals*. McGraw-Hill Science/Engineering/Math.

S. Hohmann, et al. (2000). 'Nonlinear torque control of a spark ignition engine'. In *9th IFAC Symposium on Control in Transportation System*, Germany.

M. Jankovic (2002). 'Nonlinear control in Automotive Engine Applications'. In *Proc. of the Fifteenth International Symposium on Mathematical Theory of Networks and Systems*, University of Notre Dame.

J. Karlsson & J. Fredriksson (1999). 'Cylinder-by-Cylinder Engine Models Vs Mean Value Engine Models for use in Powertrain Control Applications'. In *SAE conference*. Paper number 99P-174.

R. Le Riche & F. Guyon (1999). 'Least squares parameter estimation and the Levenberg-Marquardt algorithm : deterministic analysis, sensitivities and numerical experiments'. Tech. Rep. 041/99, INSA Rouen, France.

K. Levenberg (1944). 'A method for the solution of certain nonlinear problems in least square s'. *Quarterly of Applied Mathematics* **2**(2):164–168.

L. Ljung (1987). *System Identification: Theory for the User*. Prentice-Hall, Englewood Cliffs, New Jersey, Etats-Unis.

K. Maertens, et al. (2004). 'Engine load prediction in off-road vehicles using multi-objective nonlinear identification'. *Control Engineering Practice* **12**(5):615–624.

D. W. Marquardt (1963). 'An algorithm for least squares estimation of nonlinear parameters'. *SIAM* **11**:431–441.

M. Minoux (1986). *Mathematical programming: Theory and Algorithms*. Wiley.

H. Namba, et al. (1992). 'Vehicle acceleration control system'. European patent EP 0 230 745 B1. property of Nippondenso Co.

B. Ripley (1996). *Pattern Recognition and Neural Networks*. Cambrigde University Press.

G. Rizzoni, et al. (2003). 'Engine control using torque estimation'. Patent US 2003/0167118 A1.

T. Sano (1995). 'A substitution method for measuring engine output torque – calculation from the combustion pressure wave'. *JSAE Review* **16**(3):328.

M. Scherer, et al. (1999). 'Device for determining the engine load for an internal combustion engine'. Patent US 005889204A.

B. Schölkopf & A. Smola (2001). *Leaning with Kernels*. MIT Press.

A. J. Smola & B. Schölkopf (1998). 'A tutorial on support vector regression'. Tech. Rep. NC2-TR-1998-030, NeuroCOLT2.

V. Vapnik (1998). *Statistical Learning Theory*. Wiley.

O. Vitek, et al. (2003). '3D In-Cylinder Flow Model of Internal Combustion Engine'. In *Proceedings of Fluid Dynamics 2003*, Prague.

Y.-Y. Wang, et al. (1997). 'Event-based estimation of indicated torque for IC engines using sliding-mode observers'. *Control Engineering Practice* **5**(8):1123–1129.

D. H. Wolpert & W. G. MacReady (1995). 'No Free Lunch Theorems for Search'. Tech. Rep. SFI-TR-95-02-010, Santa Fe Institute, Santa Fe, NM.

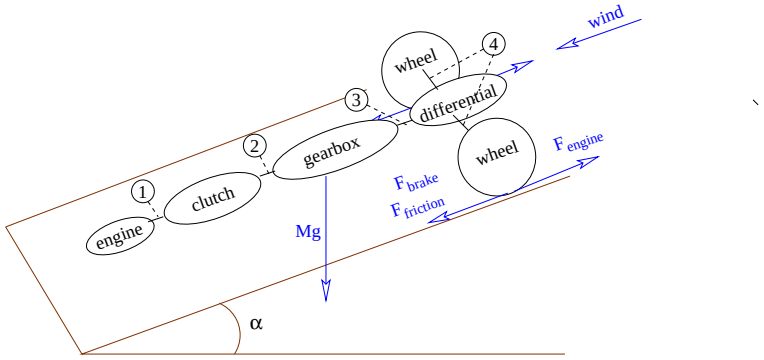

Figure 6: Sketch of a vehicle seen as a material point.

A Torque estimation and longitudinal dynamics

It is possible to propose a knowledge-based model of engine torque by applying the laws of dynamics to a car. Such a model is called external because the effects of the engine torque on the car dynamics in a given environment are modelled independently from the engine controls (accelerator position, gear, throttle, fuel flow, ...). Referring to Figure 6, the vehicle seen as a conservative material point follows the fundamental relation of dynamics

$$F_{\text{engine}} + F_{\text{gravity}} + F_{\text{friction}} + F_{\text{brake}} \;=\; (M + M_{\text{rot}})\ddot{d}\;, \tag{19}$$

where $F_{...}$ are forces, M is the vehicle mass and M_{rot} is the reduced mass of rotating elements. The right hand side of Equation (19) describes inertial effects and stems from the

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{d}}\right) \tag{20}$$

term is Lagrange's Equation, where K is the kinetic energy,

$$K \;=\; \frac{1}{2}M\dot{d}^2 + \sum_{i=1}^{4}\frac{1}{2}I_i\dot{\theta}^2\;. \tag{21}$$

To simplify the above Equation, it is considered that there are 4 pieces in rotation with moments of inertia I_i. Transmission ratios are defined here as

$$k_i \;=\; \frac{\dot{\theta}_i}{\dot{\theta}_4} \tag{22}$$

where the 4th rotating piece is the wheel of radius R. Introducing (22) into (21), the kinetic energy of the vehicle is

$$K \;=\; \frac{1}{2}(M + \sum_{i=1}^{4} I_i\frac{k_i^2}{R^2})\dot{d}^2\;. \tag{23}$$

Using this expression of K into Equation (20) yields the definition of the reduced mass of Equation (19),

$$M_{\text{rot}} \;=\; \sum_{i=1}^{4} I_i \frac{k_i^2}{R^2} \;. \tag{24}$$

Some of the forces in the left hand-side of Equation (19) are now further explicited using classical approximate relations.

$$F_{\text{engine}} \;=\; \frac{\text{torque at wheel}}{R} \;=\; \frac{P_4}{R\dot{\theta}_4} \;=\; \frac{\eta P_1}{R\dot{\theta}_4} \;=\; \frac{\eta y \dot{\theta}_1}{R\dot{\theta}_4} \;=\; \frac{k_1 \eta}{R} y \;, \tag{25}$$

where P_i is the power at the i-th rotating piece and η the driveline efficiency between the engine and the wheels. Friction forces are typically expressed as

$$F_{\text{friction}} \;=\; F_{\text{aero}} + F_{\text{road}} \;, \tag{26}$$

$$F_{\text{aero}} \;=\; \frac{1}{2}\rho S C_x (\dot{d} + \text{wind})^2 \;, \tag{27}$$

$$F_{\text{road}} \;=\; C_{\text{rf}} \dot{d}^2 \;, \tag{28}$$

where ρ is the air density, S the vehicle aerodynamic cross-section, C_x the drag coefficient and C_{rf} a road friction coefficient.

Substituting forces expressions into (19) gives a longitudinal dynamics equation involving the net engine torque y,

$$\begin{aligned} \frac{\eta y k_1}{R} - Mg\sin(\alpha) - \left(\frac{1}{2}\rho S C_x\right)(\dot{d} + \text{wind})^2 + C_{\text{rf}}\dot{d}^2 + F_{\text{brake}} \\ = \; (M + M_{\text{rot}})\ddot{d} \end{aligned} \tag{29}$$

In a conventional vehicle in real driving conditions, the road slope α, the road surface state C_{rf} and the wind are unknown. This is the reason why the external model cannot be used to estimate engine torque. Note, on the contrary, that Equation (29) could participate in estimating some of these external condition parameters.